Σχεδιασμός Εφαρμογών Διαδικτύου με Εργαλεία Ανοικτού Λογισμικού

Σημειώσεις Σεμιναρίου

Χρήση Βάσεων Δεδομένων
(MySQL)

Εισηγητής

Γιώργος Αυγέρης, MSc

Περιεχόμενα

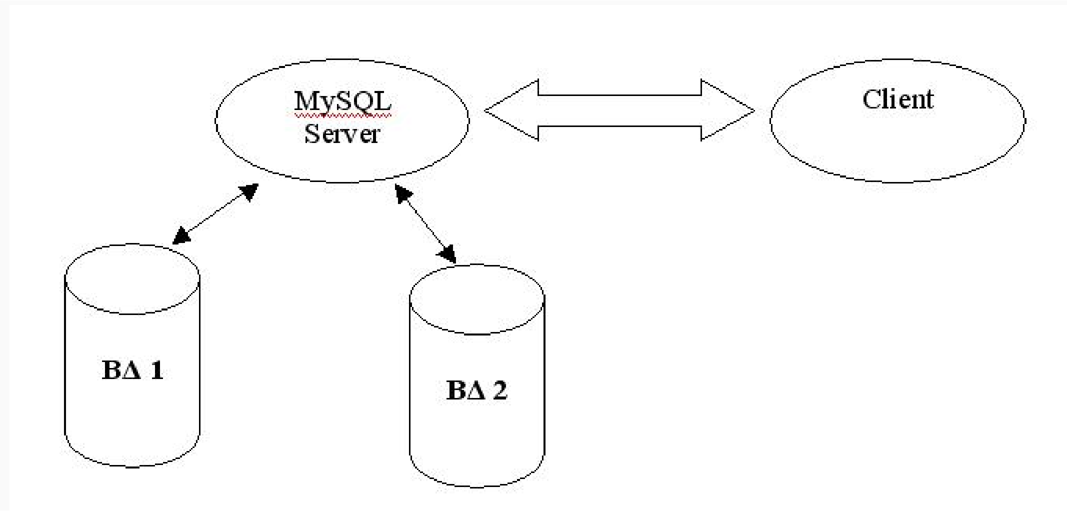
1 Χρήση Βάσεων Δεδομένων: Η MySQL 3
1.1. MySQL 3
1.2. Εγκατάσταση MySql Query Browser 4
1.3. MySQL: Σχεδιασμός βάσης δεδομένων 5
1.4. Βασικές ερωταποκρίσεις 8
1.5. Παραδείγματα χρήσης 12
2 Διασύνδεση PHP με Βάσεις Δεδομένων 14
2.1. Σύνδεση PHP-MySQL 14
2.2. Προβολή Αποτελεσμάτων Ερωταποκρίσεων 15
3 Βιβλιογραφία 17
4 Παράρτημα 18

1 Χρήση Βάσεων Δεδομένων: Η MySQL

1.1. MySQL

Η MySQL είναι ίσως μία από τις πιο δημοφιλείς ανοικτές εφαρμογές διαχείρισης βάσεων δεδομένων, η οποία έχει αναπτυχθεί και διανέμεται από την εταιρία MySQL AB

Η MySQL λειτουργεί με το μοντέλο πελάτη - εξυπηρετητή (client - server), όπου ο εξυπηρετητής διαχειρίζεται τις αιτήσεις των πελατών προς οποιαδήποτε βάση δεδομένων.



Σχήμα 1-Α

Η MySQL χρησιμοποιεί την SQL για να υλοποιήσει λειτουργίες διαχείρισης βάσεων δεδομένων, όπως η διαχείριση πινάκων (δημιουργία, τροποποίηση, διαγραφή), τη διαχείριση εγγραφών (εισαγωγή, τροποποίηση, διαγραφή) και την ανάκτηση δεδομένων από τις βάσεις, οι οποίες ικανοποιούν συγκεκριμένα κριτήρια.

Έχοντας εγκαταστήσει το λογισμικό MySQL server και MySQL client από την τοποθεσία www.mysql.com, μπορούμε να εκτελέσουμε όλες τις δυνατές λειτουργίες που μας παρέχει το σύστημα διαχείρισης βάσεων δεδομένων. Απαραίτητη προϋπόθεση για την εκτέλεση των παραπάνω λειτουργιών αποτελεί η εκκίνηση του MySQL server.

Από μία γραμμή εντολών μπορούμε να συνδεθούμε στο MySQL server, εκτελώντας την εντολή:

```
mysql -h host -u user -p,
```

όπου στη θέση του *host* πρέπει να τοποθετήσουμε το όνομα ή την IP διεύθυνση του υπολογιστή στον οποίο εκτελείται ο MySQL server. Αν εκτελούμε το πρόγραμμα πελάτη στον ίδιο υπολογιστή με τον MySQL server, μπορούμε να παραλείψουμε το τμήμα *-h host*. Το *user* πρέπει να είναι το όνομα του χρήστη που έχει οριστεί για αυτόν στη MySQL.

1.2. ΕΓΚΑΤΑΣΤΑΣΗ MYSQL QUERY BROWSER

Από την τοποθεσία www.mysql.com δίνεται η δυνατότητα να εγκαταστήσουμε γραφικά περιβάλλοντα για τη διαχείριση των βάσεων, τόσο από την πλευρά του server, όσο και από την πλευρά του client.

Ο MySQL Query Browser είναι ένα γραφικό εργαλείο, το οποίο παρέχεται από την MySQL AB, για την δημιουργία, εκτέλεση και βελτιστοποίηση αιτημάτων σε γραφικό περιβάλλον. Ενώ ο MySQL Administrator έχει σχεδιαστεί για την διαχείριση ενός MySQL εξυπηρετητή, ο MySQL Query Browser έχει σχεδιαστεί για να σας βοηθήσει να θέτετε αιτήματα και να αναλύετε δεδομένα, τα οποία είναι αποθηκευμένα στην MySQL βάση δεδομένων σας.

Ενώ όλα τα αιτήματα, που μπορούν να εκτελεστούν με τον MySQL Query Browser, μπορούν επίσης να εκτελεστούν με την εφαρμογή γραμμής εντολών **mysql**, ο MySQL Query Browser επιτρέπει την υποβολή αιτημάτων και την επεξεργασία των δεδομένων με ένα πιο διαισθητικό, γραφικό τρόπο.

Ο MySQL Query Browser είναι σχεδιασμένος, ώστε να συνεργάζεται με τις εκδόσεις 4.0 και άνω της MySQL.

Ο MySQL Query Browser εγκαθίσταται με την χρήση ενός πακέτου εγκατάστασης Windows Installer (`.msi`), το οποίο μπορεί να χρησιμοποιηθεί σε όλα τα λειτουργικά συστήματα Windows. Το πακέτο MSI περιέχεται σε ένα archive με την ονομασία `mysql-query-browser-`*`version`*`-win.msi`, όπου με *`version`* δηλώνεται η έκδοση του MySQL Query Browser.

Για να εγκαταστήσετε τον MySQL Query Browser, κάντε δεξί κλικ στο αρχείο MSI και επιλέξτε Install. Αφού το πρόγραμμα εγκατάστασης σας ζητήσει τις παραμέτρους της εγκατάστασης, η εγκατάσταση θα ξεκινήσει αυτομάτως. Κατά την διάρκεια της εγκατάστασης μπορείτε να επιλέξετε, αν θέλετε, να τοποθετηθεί μια συντόμευση στο μενού `Start` και ένα εικονίδιο στην επιφάνεια εργασίας.

Αν έχετε πρόβλημα με την εκτέλεση του προγράμματος εγκατάστασης μπορείτε, ως εναλλακτική λύση, να κατεβάσετε ένα αρχείο a ZIP file το οποίο δεν περιέχει πρόγραμμα εγκατάστασης. Αυτό το αρχείο έχει την ονομασία `mysql-query-browser-`*`version`*`-win-noinstall.zip`. Χρησιμοποιώντας ένα πρόγραμμα διαχείρισης συμπιεσμένων αρχείων, αποσυμπιέστε το σε έναν κατάλογο της επιλογής σας. Μπορείτε να δημιουργήσετε συντομεύσεις για το `MySQLQueryBrowser.exe` στην επιφάνεια εργασίας ή στην quick launch bar.

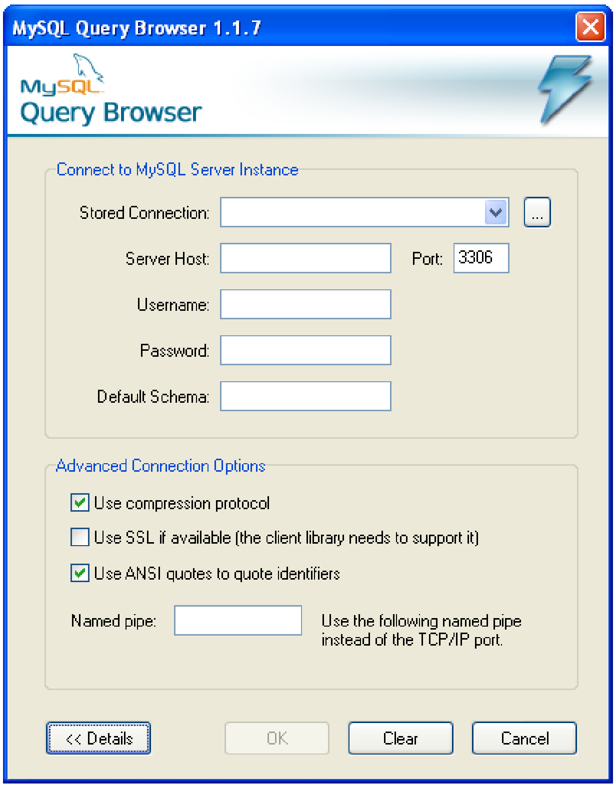
Αν δεν αλλάξετε τις επιλογές εγκατάστασης, ο MySQL Query Browser θα εγκατασταθεί στον κατάλογο:

```
C:\%PROGRAMFILES%\MySQL\MySQL Query Browser 1.1\MySQLQueryBrowser.exe,
```

όπου %PROGRAMFILES% είναι ο προεπιλεγμένος (default) κατάλογος για τα προγράμματα στο μηχάνημά σας. Για παράδειγμα αυτός ο κατάλογος μπορεί να είναι ο C:\Program Files.

Μόλις ξεκινήσει ο MySQL Query Browser, εμφανίζεται ένα παράθυρο διαλόγου (Εικόνα 1-1) το οποίο αφορά την σύνδεση. Πρέπει να ορίσετε τον MySQL εξυπηρετητή, με τον οποίο επιθυμείτε να συνδεθείτε, τα διαπιστευτήρια, τα οποία χρειάζονται για την πιστοποίησή σε αυτόν τον εξυπηρετητή, το μηχάνημα, στο οποίο τρέχει ο εξυπηρετητής (και σε ποιά θύρα ακούει) και την προεπιλεγμένη (default) βάση δεδομένων (Σχήμα), στην οποία θα θέτετε αιτήματα. Μπορείτε επιπλέον να ορίστε ένα πλήθος επιλογών, αν χρειάζεται.

Πρέπει να προεπιλέξετε (default) μια βάση δεδομένων, ώστε να θέσετε αιτήματα. Υπάρχει η δυνατότητα να επιλέξετε την προεπιλεγμένη (default) βάση δεδομένων, αφού συνδεθείτε στον εξυπηρετητή, αλλά η επιλογή της προεπιλεγμένης (default), από το παράθυρο διαλόγου σύνδεσης, μπορεί να εξοικονομήσει χρόνο στις επόμενες συνδέσεις.



Εικόνα 1-1

Αν η σύνδεση με τον εξυπηρετητή επιτευχθεί, όλες οι τιμές οι οποίες συμπληρώθηκαν στα πεδία του παραθύρου διαλόγου της σύνδεσης σώζονται για τις μελλοντικές. Ωστόσο, το πεδίο Κωδικός είναι πάντα κενό: Για λόγους ασφαλείας ο κωδικός πρόσβασης (password) δεν αποθηκεύεται μαζί με τις άλλες επιλογές, εκτός και αν εσείς το ζητήσετε ρητά στον τομέα General Options, ο οποίος είναι μέρος του παραθύρου διαλόγου Επιλογές.

Μπορείτε να αλλάξετε τις τιμές στα προφίλ σύνδεσης, απλά αντικαθιστώντας τις υπάρχουσες τιμές με καινούργιες. Ομοίως, αν επιλέξετε ένα άλλο προφίλ σύνδεσης, μπορείτε να αλλάξετε οποιαδήποτε από τις τιμές του. Όταν πατήσετε το κουμπί OK, αφού έχετε αλλάξει τις τιμές ενός προφίλ, οι αλλαγές αποθηκεύονται οριστικά, αν επιτευχθεί σύνδεση με τον MySQL εξυπηρετητή. Μπορείτε να κάνετε κλικ στο κουμπί ... δίπλα στο drop-down box Connection. Αυτό εμφανίζει ένα παράθυρο διαλόγου **Επιλογές**, το οποίο έχει έναν τομέα **Connections** για την μεταβολή του προφίλ σύνδεσης.

1.3. MySQL: Σχεδιασμός βάσης δεδομένων

Στην ενότητα αυτή, θα προσπαθήσουμε να δώσουμε μερικές πρακτικές συμβουλές και κανόνες για τη σχεδίαση μιας βάσης δεδομένων. Επισημαίνεται ξανά, ότι η διαδικασία της σχεδίασης μιας βάσης δεδομένων είναι πολύ σημαντική για τη μετέπειτα σωστή λειτουργία και αξιοποίηση της βάσεως. Για να καταδείξουμε τους κανόνες σχεδίασης, θα χρησιμοποιήσουμε ένα κοινό παράδειγμα γύρω από το περιβάλλον μιας ημερίδας επιμόρφωσης καθηγητών.

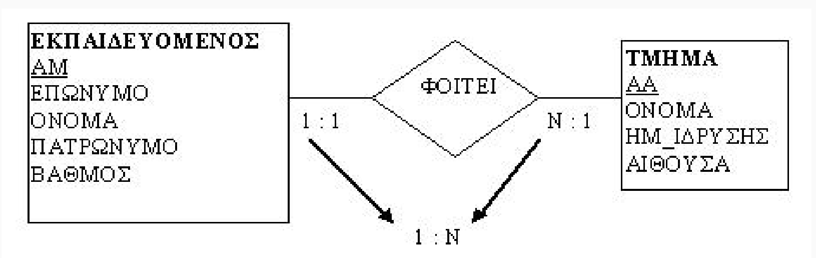
Παράδειγμα

Έστω λοιπόν ότι για κάθε τμήμα της ημερίδας επιμόρφωσης καθηγητών κρατάμε διάφορες πληροφορίες όπως όνομα, ημερομηνία ίδρυσης κλπ. Σε κάθε τμήμα, διδάσκονται διάφορα μαθήματα και φοιτούν εκπαιδευόμενοι. Για κάθε μάθημα υπάρχουν κάποιοι εκπαιδευτές που το διδάσκουν και κάποιοι εκπαιδευόμενοι που το παρακολουθούν. Κάθε διδασκαλία, δε, γίνεται κατά τη διάρκεια μίας ή περισσοτέρων διδακτικών περιόδων. Για κάθε εκπαιδευόμενο κρατάμε στοιχεία όπως ονοματεπώνυμο, ΑΜ και βαθμό.

Κατ' αρχήν, πρέπει να απεικονίσουμε την πραγματικότητα, όπως περιγράφηκε στο παράδειγμα σε λεκτική μορφή φυσικής γλώσσας, σε μια μορφή που να είναι κοντά στο σχεσιακό μοντέλο. Για το στάδιο αυτό, χρησιμοποιείται συχνά μια ενδιάμεση μορφή αναπαράστασης, επονομαζόμενη **Μοντέλο Οντοτήτων-Συσχετίσεων**, η οποία αποτελείται από δύο δομικά στοιχεία:

- **Οντότητες**, που αναπαριστούν φυσικές οντότητες του πραγματικού κόσμου, και
- **Συσχετίσεις**, που αναπαριστούν αλληλεξαρτήσεις των οντοτήτων, ή δράσεις.

Κάθε συσχέτιση έχει ένα **βαθμό πληθικότητας**, που χαρακτηρίζει τη συσχέτιση μεταξύ δύο οντοτήτων. Έχουμε τριών ειδών σχέσεις: αυτές με βαθμό πληθικότητας 1:1, αυτές με βαθμό πληθικότητας 1:Ν και αυτές με βαθμό πληθικότητας Μ:Ν. Οι πρώτες ονομάζονται και συσχετίσεις «ένα προς ένα», οι δεύτερες συσχετίσεις «ένα προς πολλά» και οι τρίτες «πολλά προς πολλά». Ο βαθμός πληθικότητας μιας συσχέτισης προκύπτει σαν συνδυασμός των επί μέρους πληθικοτήτων των στοιχείων της. Ας δούμε ένα παράδειγμα:



Σχήμα 1-Β

Βλέπουμε, ότι στη γραμμή που συνδέουμε τη συσχέτιση ΦΟΙΤΕΙ με την οντότητα ΤΜΗΜΑ έχουμε σημειώσει πληθικότητα 1:Ν. Αυτό εξηγείται ως εξής: «Σε ένα (1) τμήμα φοιτούν πολλοί (Ν) εκπαιδευόμενοι». Στην άλλη μεριά της συσχέτισης, βλέπουμε τη σύνδεση με την οντότητα ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ. Παρατηρούμε ότι η συσχέτιση έχει πληθικότητα 1:1, καθώς «Ένας (1) εκπαιδευόμενος, φοιτεί σε ένα (1) τμήμα». Ο συνολικός βαθμός της συσχέτισης είναι 1:Ν, όπως προκύπτει από το συνδυασμό των επί μέρους πληθικοτήτων. Η οντότητα που συμμετέχει στη συσχέτιση με επιμέρους σχέση 1:Ν ονομάζεται πρωτεύουσα οντότητα (εδώ: ΤΜΗΜΑ) και η οντότητα που συμμετέχει στη συσχέτιση με επιμέρους σχέση 1:1 ονομάζεται οντότητα στόχος (εδώ: ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ).

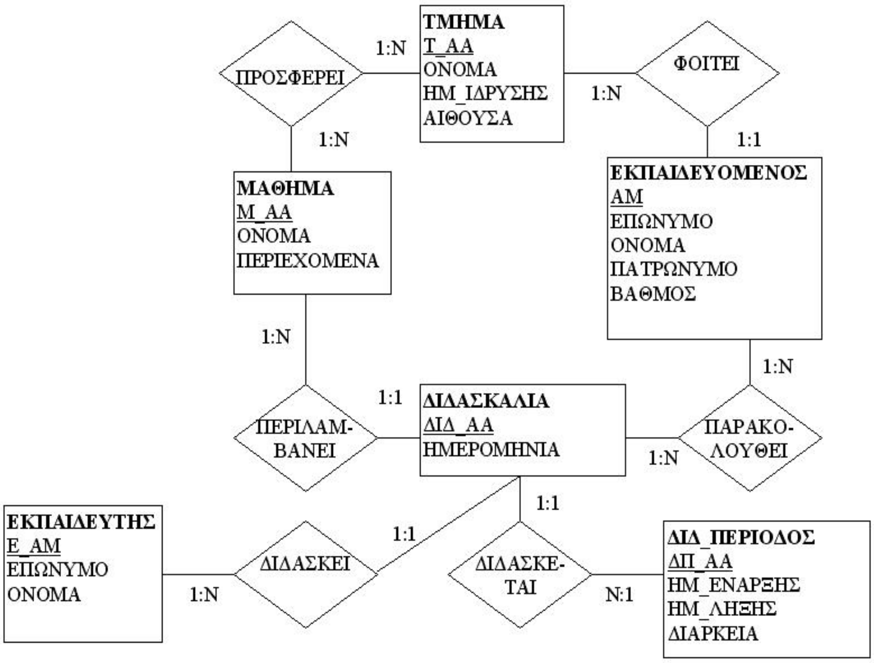
Πάντα διαβάζουμε (και ζωγραφίζουμε) από την οντότητα προς τη συσχέτιση. Πάντα την κρίσιμη πρόταση την ξεκινάμε με «ένας/μία». Κάθε φορά δηλαδή, παίρνουμε μία οντότητα από τη μια άκρη της συσχέτισης και βλέπουμε με πόσες οντότητες από την άλλη άκρη της συσχέτισης σχετίζεται. Αυτό το κάνουμε και από τις δύο μεριές. Ο συνολικός βαθμός προκύπτει αν αγνοήσουμε τις επί μέρους υποθέσεις «ένας/μία» (όπως φαίνεται άλλωστε και στο σχήμα).

Συμβουλές:

- Προσπαθήστε να παραμείνετε όσο το δυνατό πιο κοντά στην λεκτική περιγραφή. Μην προσπαθήσετε να περάσετε τις όποιες προγραμματιστικές υποθέσεις ή επεμβάσεις από τώρα στη σχεδίαση: αυτό θα γίνει σε επόμενο βήμα.
- Μια καλή αρχή για τον εντοπισμό οντοτήτων και συσχετίσεων είναι τα ουσιαστικά και τα ρήματα του κειμένου της λεκτικής περιγραφής των προδιαγραφών. Τα ουσιαστικά είναι συνήθως καλοί υποψήφιοι για οντότητες και τα ρήματα για συσχετίσεις. Φυσικά, η συμβουλή δεν αποτελεί απόλυτο κανόνα, καθώς υπάρχουν πάντα ισοδύναμες εκφράσεις για το ίδιο πράγμα (π.χ. «φοιτεί» και «φοίτηση» μπορεί να είναι όροι που χρησιμοποιούνται για να εκφράσουν το ίδιο πράγμα).

- Θυμηθείτε να σημειώσετε πρωτεύοντα κλειδιά (ή ακόμα και να εισάγετε αν χρειάζεται) από νωρίς.
- Όλες οι συσχετίσεις είναι πρώτης τάξεως υποψήφιοι για συσχετίσεις εξωτερικών κλειδιών. Θυμηθείτε να σημειώσετε τους βαθμούς πληθικότητάς τους.

Στο παράδειγμά μας, το μοντέλο Οντοτήτων-Συσχετίσεών του φαίνεται στο παρακάτω σχήμα:



Σχήμα 1-C

Κανόνες κατασκευής σχήματος βάσης δεδομένων από το μοντέλο οντοτήτων συσχετίσεων

- Κάθε οντότητα απεικονίζεται σε ένα πίνακα.
- Για κάθε συσχέτιση που και οι δύο ακμές της είναι πληθικότητας 1:Ν (έχει συνολικό βαθμό πληθικότητας, δηλαδή, Μ:Ν), κατασκευάζουμε και τον αντίστοιχο πίνακα. Ο πίνακας περιλαμβάνει ως πεδία τα πρωτεύοντα κλειδιά των οντοτήτων που εμπλέκονται στη συσχέτιση.
- Για κάθε συσχέτιση που η μία ακμή της είναι πληθικότητας 1:Ν και η άλλη 1:1 (έχει συνολικό βαθμό πληθικότητας, δηλαδή, 1:Ν), προσθέτουμε στην οντότητα που συνδέεται με πληθικότητα 1:1, ένα πεδίο που είναι εξωτερικό κλειδί στον πίνακα της άλλης οντότητας.
- Στις συσχετίσεις που και οι δύο ακμές τους είναι πληθικότητας 1:1 (με συνολικό βαθμό πληθικότητας, δηλαδή, 1:1) κάνουμε το ίδιο, χωρίς να έχει ιδιαίτερη σημασία το πού τοποθετείται το επιπλέον πεδίο.
- Φροντίζουμε για κάθε πίνακα να υπάρχει πρωτεύον κλειδί. Φροντίζουμε ώστε όλα τα εξωτερικά κλειδιά να είναι δηλωμένα επίσης.
- Για κάθε πεδίο κάθε πίνακα καθορίζουμε τον τύπο του (αριθμός, κείμενο, ημερομηνία, κλπ.). Επιπλέον, μπορούμε να καθορίσουμε περιορισμούς για τον αν επιτρέπεται κάποια εγγραφή να μην έχει τιμή στο εν λόγω πεδίο.

__ΔΕΝ__ πρέπει να ξεχνάμε ποτέ, σε ποια μεριά της συσχέτισης 1:Ν πρέπει να γίνει η προσθήκη του επιπλέον πεδίου. Ο κανόνας είναι ξεκάθαρος: το επιπλέον πεδίο πηγαίνει στον πίνακα στόχο.

Προσέξτε ιδιαίτερα την ονοματολογία των πεδίων (αλλά και των πινάκων). Δύο χρήσιμες συμβουλές είναι:

- Τα ονόματα που δίνονται να είναι εκφραστικά (εις βάρος της ταχύτητας πληκτρολόγησης εν ανάγκη).
- Το όνομα του πίνακα να προηγείται του ονόματος του πεδίου (εν ανάγκη και με μια μικρή σύντμηση).

1.4. Βασικες ερωταποκρισεις

Η SQL (Structured Query Language) είναι σήμερα η πιο δημοφιλής και πιο διαδεδομένη γλώσσα ανάπτυξης και διαχείρισης σχεσιακών βάσεων δεδομένων. Η SQL αποτελείται από εντολές με τα ορίσματά τους, τις οποίες μπορούμε να χρησιμοποιήσουμε με συγκεκριμένους κανόνες σύνταξης για να πάρουμε τα αποτελέσματα που θέλουμε.

Με την SQL μπορούμε:

- να δημιουργήσουμε μια βάση δεδομένων και τους πίνακές της με τα αντίστοιχα πεδία
- να εισάγουμε εγγραφές στους πίνακες
- να τροποποιήσουμε και να διαγράψουμε εγγραφές
- να αλλάξουμε τη δομή των πινάκων με προσθήκη και διαγραφή πεδίων
- να εμφανίσουμε πληροφορίες

Οι Τύποι Δεδομένων της SQL

Η SQL υποστηρίζει τους εξής τύπους δεδομένων για τα πεδία ενός πίνακα:

- char(n), ένα αλφαριθμητικό (string) με n ακριβώς χαρακτήρες.
- varchar(n), ένα αλφαριθμητικό (string) με μεταβλητό μήκος, το πολύ **n** χαρακτήρες.
- int, ακέραιος αριθμός.
- smallint, ακέραιος αριθμός με μικρές τιμές.
- Numeric(p, d), αριθμός με p ψηφία, από τα οποία τα d είναι δεκαδικά.
- real, αριθμός κινητής υποδιαστολής απλής ακρίβειας.
- double precision, αριθμός κινητής υποδιαστολής διπλής ακρίβειας.
- float(n), αριθμός κινητής υποδιαστολής με ακρίβεια n ψηφίων.
- date, ημερομηνία (ημέρα, μήνας, έτος).
- time, ώρα (ώρα, λεπτά, δευτερόλεπτα).

Οι εντολές της SQL κατηγοριοποιούνται σε 2 κατηγορίες

- Εντολές DML (Data Manipulation Language)
- Εντολές DDL (Data Definition Language)

Η πρώτη κατηγορία περιλαμβάνει εντολές διαχείρισης δεδομένων όπως οι εντολές: ***SELECT, INSERT, UPDATE, DELETE*** ενώ η δεύτερη κατηγορία περιλαμβάνει εντολές διαχείρισης του σχήματος της βάσης όπως οι εντολές ***CREATE, ALTER***.

DML - Εντολή SELECT

Η βασική εντολή για την ανάκτηση πληροφοριών από τη ΒΔ είναι η εντολή SELECT. Η βασική μορφή της εντολής είναι η εξής:

SELECT <λίστα πεδίων>

FROM <λίστα πινάκων>

WHERE <συνθήκη>;

Στη <λίστα πεδίων> δηλώνουμε τα πεδία που θα εμφανίζονται στο αποτέλεσμα. Αν θέλουμε να εμφανίζονται όλα τα πεδία θα πρέπει να χρησιμοποιήσουμε τον χαρακτήρα *.

Η πρόταση **FROM** δηλώνει τους πίνακες στους οποίους θα γίνει η αναζήτηση.

Η πρόταση **WHERE** είναι προαιρετική και περιέχει τη συνθήκη που θέλουμε να ικανοποιούν οι εγγραφές του αποτελέσματος.

Παράδειγμα

Η παρακάτω εντολή SELECT εμφανίζει το όνομα, το επώνυμο και το βαθμό για όλους τους εκπαιδευόμενους.

```
SELECT ΟΝΟΜΑ, ΕΠΩΝΥΜΟ, ΒΑΘΜΟΣ

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ;
```

Πρόταση ORDER BY

Με την πρόταση **ORDER BY** μπορούμε να ταξινομήσουμε το αποτέλεσμα μιας εντολής **SELECT** ως προς κάποιο πεδίο. Για παράδειγμα, να εμφανίσουμε τη λίστα των εκπαιδευόμενων ταξινομημένη ως προς επώνυμο, χρησιμοποιούμε την πρόταση **ORDER BY** ως εξής:

```
SELECT *

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ

ORDER BY ΕΠΩΝΥΜΟ;
```

Εξ ορισμού η ταξινόμηση γίνεται κατ' αύξουσα σειρά. Μπορούμε να χρησιμοποιήσουμε και τους όρους **ASC** για αύξουσα και **DESC** για φθίνουσα ταξινόμηση, ως εξής :

```
SELECT *

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ

ORDER BY ΕΠΩΝΥΜΟ ASC;
```

Συναρτήσεις Ομαδοποίησης

Η SQL χρησιμοποιεί μερικές πολύ χρήσιμες συναρτήσεις, που ονομάζονται συναρτήσεις ομαδοποίησης (**aggregate functions**) και οι οποίες δέχονται ένα σύνολο τιμών και επιστρέφουν μία τιμή. Οι συναρτήσεις αυτές είναι οι εξής :

- Count (Απαρίθμηση)
- Sum (Άθροισμα)
- Avg (Μέσος Όρος)
- Max (Μέγιστο)
- Min (Ελάχιστο)

Οι συναρτήσεις Sum και Avg εργάζονται μόνο με αριθμητικές τιμές, ενώ οι υπόλοιπες συναρτήσεις μπορούν να δεχθούν και αλφαριθμητικές τιμές.

Παράδειγμα

Για να βρούμε τον συνολικό αριθμό των εκπαιδευομένων πρέπει να κατασκευάσουμε το εξής ερώτημα:

```
SELECT COUNT(*)

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ;
```

Το αποτέλεσμα θα είναι ένας μόνο αριθμός, δηλ. ο συνολικός αριθμός των εκπαιδευόμενων, δηλαδή το σύνολο των εγγραφών του πίνακα ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ.

Η πρόταση GROUP BY

Με την πρόταση **GROUP BY** μπορούμε να ξεχωρίσουμε (απομονώσουμε) τις εγγραφές ενός πίνακα σε ανεξάρτητα υποσύνολα και μετά να εφαρμόσουμε μια συνάρτηση ομαδοποίησης σε κάθε υποσύνολο.

Παράδειγμα

Θέλουμε να κάνουμε καταμέτρηση των εκπαιδευτικών ανά βαθμό. Αυτό μπορεί να πραγματοποιηθεί ομαδοποιώντας τους εκπαιδευομένους ανά βαθμό και με τη χρήση της συνάρτησης **COUNT** να βρούμε το πλήθος της κάθε ομάδας.

```
SELECT ΒΑΘΜΟΣ, COUNT(*)

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ

GROUP BY ΒΑΘΜΟΣ;
```

Κάθε εγγραφή του αποτελέσματος θα περιέχει ένα πεδίο για το βαθμό και ένα πεδίο για το πλήθος των εγγραφών του πίνακα με το συγκεκριμένο βαθμό.

Η πρόταση HAVING

Η πρόταση **HAVING** χρησιμοποιείται στις συνθήκες που εφαρμόζονται σε κάθε υποσύνολο που δημιουργείται από την πρόταση **GROUP BY** και όχι σε κάθε εγγραφή, όπως συμβαίνει με τη συνθήκη της πρότασης **WHERE**.

Παράδειγμα

Εάν επιθυμούσαμε να εμφανίσουμε τους εκπαιδευόμενους που έχουν Μ.Ο. βαθμού πάνω από 10 θα έπρεπε να χρησιμοποιήσουμε την πρόταση **HAVING**.

```
SELECT ΟΝΟΜΑ, AVG(ΒΑΘΜΟΣ)

FROM ΒΑΘΜΟΙ_ΕΚΠΑΙΔΕΥΟΜΕΝΟΥ

GROUP BY ΟΝΟΜΑ

HAVING AVG(ΒΑΘΜΟΣ) > 10 ;
```

Πράξεις Συνόλων

Επειδή οι σχέσεις αντιστοιχούν σε σύνολα, μπορούμε να χρησιμοποιήσουμε τις πράξεις μεταξύ συνόλων και στις σχέσεις. Μπορούμε να υλοποιήσουμε την πράξη της τομής, της ένωσης αλλά και της διαφοράς, ανάμεσα σε δύο σχέσεις, αρκεί να έχουν τα ίδια πεδία ή πεδία με το ίδιο πεδίο ορισμού.

Παράδειγμα

Εάν για παράδειγμα έχουμε δύο πίνακες έναν για διοικητικό προσωπικό και έναν για καθηγητές και θέλουμε να πάρουμε τα ονόματά τους και τα τηλέφωνά τους θα γράφαμε:

```
SELECT ΟΝΟΜΑ, ΤΗΛΕΦΩΝΟ

FROM ΔΙΟΙΚΗΤΙΚΟ_ΠΡΟΣΩΠΙΚΟ

UNION

SELECT ΟΝΟΜΑ, ΤΗΛΕΦΩΝΟ

FROM ΚΑΘΗΓΗΤΕΣ
```

DML - Εντολή INSERT

Με την εντολή **INSERT** μπορούμε να καταχωρήσουμε πλειάδες-εγγραφές σε μια έναν υπάρχον πίνακα. Η σειρά των πεδίων και οι αντίστοιχοι τύποι δεδομένων που αυτά φιλοξενούν (κείμενο, αριθμός, κλπ) πρέπει υποχρεωτικά να τηρηθούν. Για τα πεδία που δεν έχουμε κάποια τιμή, μπορούμε να δηλώσουμε τιμή null, εφόσον φυσικά αυτό επιτρέπεται.

Παράδειγμα

```
INSERT INTO ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ VALUES (1, ‘ΕΥΑΓΓΕΛΟΥ', ‘ΓΕΩΡΓΙΟΣ', ‘ΕΥΑΓΓΕΛΟΣ', ‘Α');
```

Αν θέλουμε να εισάγουμε μια εγγραφή αλλά δεν θέλουμε να βάλουμε τιμές σε όλα τα γνωρίσματά του τότε χρησιμοποιούμε την σύνταξη:

```
INSERT INTO ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ(Α_Μ, ΕΠΩΝΥΜΟ, ΟΝΟΜΑ, ΠΑΤΡΩΝΥΜΟ)
VALUES (1, ‘ΕΥΑΓΓΕΛΟΥ', ‘ΓΕΩΡΓΙΟΣ', ‘ΕΥΑΓΓΕΛΟΣ ');
```

Εδώ δεν έχουμε βάλει τιμή στο πεδίο ΒΑΘΜΟΣ οπότε δεν το έχουμε συμπεριλάβει στη λίστα πεδίων.

DML - Εντολή DELETE

Με την εντολή **DELETE FROM** μπορούμε να διαγράψουμε ολόκληρες εγγραφές και όχι μεμονωμένα πεδία (στήλες). Για παράδειγμα, μπορούμε να διαγράψουμε όλους τους εκπαιδευόμενους με βαθμό Α, ως εξής:

Παράδειγμα

```
DELETE FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ
WHERE ΒΑΘΜΟΣ = ‘Α';
```

DML - Εντολή Update

Με την εντολή **UPDATE** μπορούμε να τροποποιήσουμε την τιμή κάποιων πεδίων από ορισμένες ή και απ' όλες τις εγγραφές ενός πίνακα. Παραδείγματος χάρη, να αλλάξουμε το βαθμό των εκπαιδευόμενων από Β σε Α:

Παράδειγμα

```
UPDATE ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ
SET ΒΑΘΜΟΣ = ‘Α'
WHERE ΒΑΘΜΟΣ= ‘Β';
```

DDL - Εντολή CREATE TABLE

Για να δημιουργήσουμε έναν πίνακα χρησιμοποιούμε την εντολή **CREATE TABLE**. Μετά την εντολή **CREATE TABLE** γράφουμε το όνομα του πίνακα και ακολουθεί μια παρένθεση που περιέχει τα ονόματα των πεδίων με τους τύπους δεδομένων τους.

Για παράδειγμα, για να δημιουργήσουμε τον πίνακα ΤΜΗΜΑ δίνουμε την εξής εντολή :

```
CREATE TABLE ΤΜΗΜΑ
(       ΤΜ_ΑΑ           int not null,
        ΟΝΟΜΑ           char(20),
        ΗΜ_ΙΔΡΥΣΗΣ      date,
        ΑΙΘΟΥΣΑ         char(10),
        PRIMARY KEY (ΤΜ_ΑΑ)
);
```

DDL - Εντολή ALTER TABLE

Σε περιπτώσεις που χρειάζεται να γίνουν αλλαγές πάνω στο σχήμα του πίνακα (π.χ. προσθήκη πεδίου, αλλαγή τύπου πεδίου, προσθήκη ξένου κλειδιού κλπ) χρησιμοποιούμε την εντολή **ALTER TABLE.**

Παράδειγμα

Για να προσθέσουμε ένα πεδίο στον πίνακα ΤΜΗΜΑ εκτελούμε την εντολή:

```
ALTER TABLE ΤΜΗΜΑ ADD (ΔΙΕΥΘΥΝΣΗ CHAR(20));
```

Τα καινούργια πεδία που προστίθενται σε μια πίνακα, έχουν αρχικά τιμές ίσες με null.

Ακεραιότητα Αναφορών

Για να δηλώσουμε ένα ξένο κλειδί και την αναφορά του, χρησιμοποιούμε τους όρους **FOREIGN KEY** και **REFERENCES** όταν δημιουργούμε έναν πίνακα:

```
CREATE TABLE ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ
(       AM              int not null,
        TM_AA           int not null,
        ΕΠΩΝΥΜΟ         char(20),
        ΟΝΟΜΑ           char(20),
        ΠΑΤΡΩΝΥΜΟ       char(20),
        ΒΑΘΜΟΣ          char(20),
        PRIMARY KEY (AM),
        FOREIGN KEY (TM_AA) REFERENCES ΤΜΗΜΑ(TM_AA)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

Για να μην έχουμε παραβίαση της ακεραιότητας αναφοράς, χρησιμοποιούμε τους όρους **ON DELETE CASCADE** και **ON UPDATE CASCADE** αντίστοιχα.

1.5. ΠΑΡΑΔΕΙΓΜΑΤΑ ΧΡΗΣΗΣ

- Σε περίπτωση που επιθυμούμε να μην εμφανίζονται διπλότυπες εγγραφές στο αποτέλεσμα, δηλαδή οι εγγραφές με ίδιες τιμές στα συγκεκριμένα πεδία, χρησιμοποιούμε τον όρο DISTINCT, ως εξής :

```
SELECT DISTINCT ΟΝΟΜΑ, ΕΠΩΝΥΜΟ, ΒΑΘΜΟΣ

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ;
```

- Αν θέλουμε να δούμε όλα τα πεδία των εκπαιδευομένων, θα πρέπει να χρησιμοποιήσουμε το σύμβολο *, ως εξής :

```
SELECT *

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ;
```

- Η επόμενη εντολή εμφανίζει εκείνους τους εκπαιδευόμενους με ΑΜ μεγαλύτερο από μια συγκεκριμένη τιμή :

```
SELECT *

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ

WHERE AM > 10;
```

Παραδείγματα χρήσης

- Εάν επιθυμούσαμε να χρησιμοποιήσουμε ένα πιο σύνθετο κριτήριο στο ερώτημα, θα μπορούσαμε στην πρόταση **WHERE** να χρησιμοποιήσουμε κάποιον από τους λογικούς τελεστές **OR** ή **AND**. Για παράδειγμα εάν θέλαμε να εμφανίσουμε τους εκπαιδευόμενους με ΑΜ μεταξύ δύο τιμών θα μπορούσαμε να χρησιμοποιήσουμε το εξής ερώτημα:

```
SELECT *

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ

WHERE AM > 10 AND AM < 20;
```

- Εάν επιθυμούσαμε να ανακτήσουμε δεδομένα από πεδία που περιέχονται σε περισσότερους από έναν πίνακες, οι οποίοι συσχετίζονται, όπως για παράδειγμα να βρούμε το όνομα, το επώνυμο των εκπαιδευόμενων και την αίθουσα του τμήματος στην οποία παρακολουθούν τη διδασκαλία οι εκπαιδευόμενοι θα κατασκευάζαμε το εξής ερώτημα:

```
SELECT ΕΠΩΝΥΜΟ, ΟΝΟΜΑ, ΑΙΘΟΥΣΑ

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ, ΤΜΗΜΑ

WHERE ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ.ΤΜ_ΑΑ= ΤΜΗΜΑ.ΤΜ_ΑΑ;
```

- Εάν θέλουμε να κάνουμε καταμέτρηση των εκπαιδευτικών που έχουν βαθμό Α, Β ή Γ ανά βαθμό θα κατασκευάζαμε το εξής ερώτημα:

```
SELECT ΒΑΘΜΟΣ, COUNT(*)

FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ

WHERE ΒΑΘΜΟΣ IN (‘Α’, ‘Β’, ‘Γ’)

GROUP BY ΒΑΘΜΟΣ;
```

- Για να διαγράψουμε όλες τις εγγραφές του πίνακα ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ χρησιμοποιούμε την εντολή:

```
DELETE FROM ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ
```

- Για να κάνουμε NULL το βαθμό όλων των εκπαιδευόμενων, πρέπει να κατασκευάσουμε το εξής ερώτημα:

```
UPDATE ΕΚΠΑΙΔΕΥΟΜΕΝΟΣ

SET ΒΑΘΜΟΣ =NULL;
```

- Για να τροποποιήσουμε έναν πίνακα διαγράφοντας ένα πεδίο, δίνουμε την εντολή ALTER TABLE και το όρισμα DROP, ως εξής:

```
ALTER TABLE ΤΜΗΜΑ DROP ΔΙΕΥΘΥΝΣΗ;
```

- Για να διαγράψουμε τελείως έναν πίνακα από τη βάση δεδομένων στην οποία ανήκει, μαζί με τις τιμές των εγγραφών του, δίνουμε την εντολή DROP TABLE, ως εξής :

```
DROP TABLE ΤΜΗΜΑ;
```

2 Διασύνδεση PHP με Βάσεις Δεδομένων

2.1. Σύνδεση PHP-MySQL

Πριν μπορέσουμε να πάρουμε το περιεχόμενο μιας βάσης δεδομένων της MySQL για να το εισάγουμε σε μια ιστοσελίδα, πρέπει πρώτα να δούμε πώς θα καθιερώσουμε μια σύνδεση (connection) με την MySQL. Η κλήση στην ακόλουθη συνάρτηση της PHP δημιουργεί αυτή τη σύνδεση :

```
mysql_connect(<address>, <username>, <password>);
```

Όπου το **<address>** είναι η IP διεύθυνση ή το hostname του υπολογιστή στον οποίο εκτελείται το λογισμικό του MySQL Server, που είναι το localhost αν εκτελείται στον ίδιο υπολογιστή με το λογισμικό του Web server, και τα **<username>** και **<password>** είναι τα ίδια με τα username και password της MySQL που χρησιμοποιήσαμε για να συνδεθούμε στον MySQL Server.

Η συνάρτηση **mysql_connect()** επιστρέφει έναν αριθμό για να μπορούμε να αναγνωρίσουμε (ξεχωρίσουμε) τη σύνδεση (connection). Μπορούμε να κρατήσουμε αυτή την τιμή σε μια μεταβλητή, ως εξής :

```
$dbcnx = mysql_connect("localhost", "root", "mypasswd");
```

Η τιμή που επιστρέφει η συνάρτηση **mysql_connect(),** που θα την αποκαλούμε **connection identifier** (αναγνωριστικό σύνδεσης), αποθηκεύεται σε μια μεταβλητή με όνομα **$dbcnx**. Όταν ο MySQL Server δεν είναι διαθέσιμος, η συνάρτηση **mysql_connect()** επιστρέφει την τιμή false. Έχουμε έτσι τη δυνατότητα να αντιδράσουμε σε τέτοιες καταστάσεις χρησιμοποιώντας μια εντολή if, ως εξής:

```
$dbcnx = mysql_connect("localhost", "root", "mypasswd");

if (!$dbcnx) {

        echo("<P>Η σύνδεση με τον " ."database server είναι αδύνατη</P>");

        exit();

}
```

Το επόμενο βήμα αφότου έχουμε καθιερώσει μια σύνδεση είναι να επιλέξουμε τη βάση δεδομένων με την οποία θέλουμε να δουλέψουμε. Προς τον σκοπό αυτό χρησιμοποιούμε μια άλλη συνάρτηση, ως εξής:

```
if (! mysql_select_db("SEMINAR") ) {
        echo( "<P>Αδύνατος ο εντοπισμός της " ."database SEMINAR </P>" );
        exit();
}
```

Χρησιμοποιούμε τη μεταβλητή **$dbcnx** που περιέχει το αναγνωριστικό σύνδεσης της βάσης δεδομένων (database connection identifier) για να ενημερώσουμε τη συνάρτηση ποια σύνδεση βάσης δεδομένων να χρησιμοποιήσει. Αυτή είναι μια προαιρετική παράμετρος και όταν παραλείπεται, η συνάρτηση θα χρησιμοποιήσει αυτόματα το αναγνωριστικό σύνδεσης (link identifier) της τελευταίας σύνδεσης που έχει ανοιχθεί. Αυτή η συνάρτηση επιστρέφει true όταν είναι επιτυχής και false αν συμβεί κάποιο λάθος.

2.2. ΠΡΟΒΟΛΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΕΡΩΤΑΠΟΚΡΙΣΕΩΝ

Στην PHP ο μηχανισμός για την εκτέλεση επερωτήσεων είναι η συνάρτηση **mysql_query().**

```
mysql_query(<query>, <connection id>);
```

Όπου το **<query>** είναι ένα string που περιέχει τις εντολές της SQL που θα εκτελεσθούν. Όπως και με τη συνάρτηση **mysql_select_db(),** η παράμετρος **<connection id>** (αναγνωριστικό σύνδεσης) είναι προαιρετική.

Η τιμή επιστροφής αυτής της συνάρτησης εξαρτάται από το είδος του ερωτήματος που στέλνεται. Για τις περισσότερες εντολές της SQL, η συνάρτηση **mysql_query()** επιστρέφει true ή false για να δείξει την επιτυχία ή την αποτυχία της αντίστοιχα. Για τα ερωτήματα SELECT αυτό δεν είναι αρκετό. Και αυτό γιατί εκτός από την ένδειξη αν το ερώτημα πέτυχε ή απέτυχε, η PHP πρέπει επίσης να λάβει υπόψη και τα αποτελέσματα του ερωτήματος.

Έτσι, όταν επεξεργαζόμαστε ένα ερώτημα SELECT, η συνάρτηση **mysql_query()** επιστρέφει έναν αριθμό που δείχνει ένα σύνολο αποτελεσμάτων (result set), που περιέχει μια λίστα όλων των γραμμών (rows) ή καταχωρήσεων (entries) που επιστρέφονται από το ερώτημα.

```
$result = mysql_query("SELECT NAME FROM STUDENTS");
if (!$result) {
	echo("<P> Λάθος στην εκτέλεση του ερωτήματος (query) : " .
	mysql_error() . "</P>");
	exit();
}
```

Αν δεν συμβεί κάποιο λάθος στην επεξεργασία του ερωτήματος, ο παραπάνω κώδικας θα τοποθετήσει στη μεταβλητή **$result** ένα σύνολο αποτελεσμάτων (resultset) που θα περιέχει το κείμενο όλων των ονομάτων που είναι αποθηκευμένα στον πίνακα STUDENTS.

Για να επεξεργαστούμε τις γραμμές (rows) του συνόλου αυτού μία κάθε φορά, μπορούμε να χρησιμοποιήσουμε τον βρόχο while, ως εξής :

```
while ( $row = mysql_fetch_array($result) ) {
	// επεξεργασία της γραμμής (row)
}
```

Η συνάρτηση **mysql_fetch_array()** δέχεται ένα σύνολο αποτελεσμάτων (resultset) σαν παράμετρο, που είναι αποθηκευμένο στη μεταβλητή **$result** στη συγκεκριμένη περίπτωση, και επιστρέφει την επόμενη γραμμή του result set σαν έναν πίνακα (array). Όταν δεν υπάρχουν άλλες γραμμές στο result set, η συνάρτηση **mysql_fetch_array()** επιστρέφει την τιμή false.

Αν θελήσουμε να εκτυπώσουμε τα ονόματα όλων των φοιτητών της βάσης δεδομένων, ο βρόχος while θα πρέπει να είναι ως εξής :

```
while ( $row = mysql_fetch_array($result) ) {
	echo("<P>" . $row["NAME"] . "</P>");
}
```

Παράδειγμα

Στη συνέχεια παρατίθεται ο ολοκληρωμένος κώδικας για την εκτέλεση query σε ΒΔ MySQL. Αποθηκεύστε τον παρακάτω κώδικα σε ένα αρχείο με όνομα students.php και δημοσιεύστε το σ τον Web Server.

```
<HTML>
<HEAD>
        <TITLE> Τα ονόματα των φοιτητών </TITLE>
<HEAD>
<BODY>

<?php
        // Σύνδεση με τον database server
        $dbcnx = @mysql_connect("localhost", "root", "mypasswd");
        if (!$dbcnx) {
                echo( "<P> Αδύνατη η σύνδεση με τον database server.</P>");
        exit();
}

        mysql_query("SET NAMES 'greek'", $dbcnx);
        // Επιλογή της βάσης δεδομένων SEMINAR
             if (! @mysql_select_db("SEMINAR") ) {
                 echo( "<P> Αδύνατος ο εντοπισμός της βάσης δεδομένων SEMINAR </P>" );
                exit();
             }
?>

        <P> Τα ονόματα των φοιτητών της βάσης δεδομένων είναι τα εξής : </P>
       <BLOCKQUOTE>
<?php
             // Αναζήτηση των ονομάτων
             $result = mysql_query(
             "SELECT NAME FROM STUDENTS");
             if (!$result) {
                         echo("<P> Λάθος στην εκτέλεση του ερωτήματος (query) : " .
                         mysql_error() . "</P>");
                         exit();
             }
             // Εμφάνιση του ονόματος του κάθε φοιτητή σε μια παράγραφο
             while ( $row = mysql_fetch_array($result) ) {
                         echo("<P>" . $row["NAME"] . "</P>");
              }
?>
             </BLOCKQUOTE>
</BODY>
</HTML>
```

3 Βιβλιογραφία

1. **MySQL AB.** MySQL Documentation. [Online] http://dev.mysql.com/doc/query-browser/el/index.html.

2. **Παναγιώτης Βασιλειάδης, Παναγιώτης Δομουχτσίδης.** *Συστήματα Διαχείρισης Βάσεων Δεδομένων.*

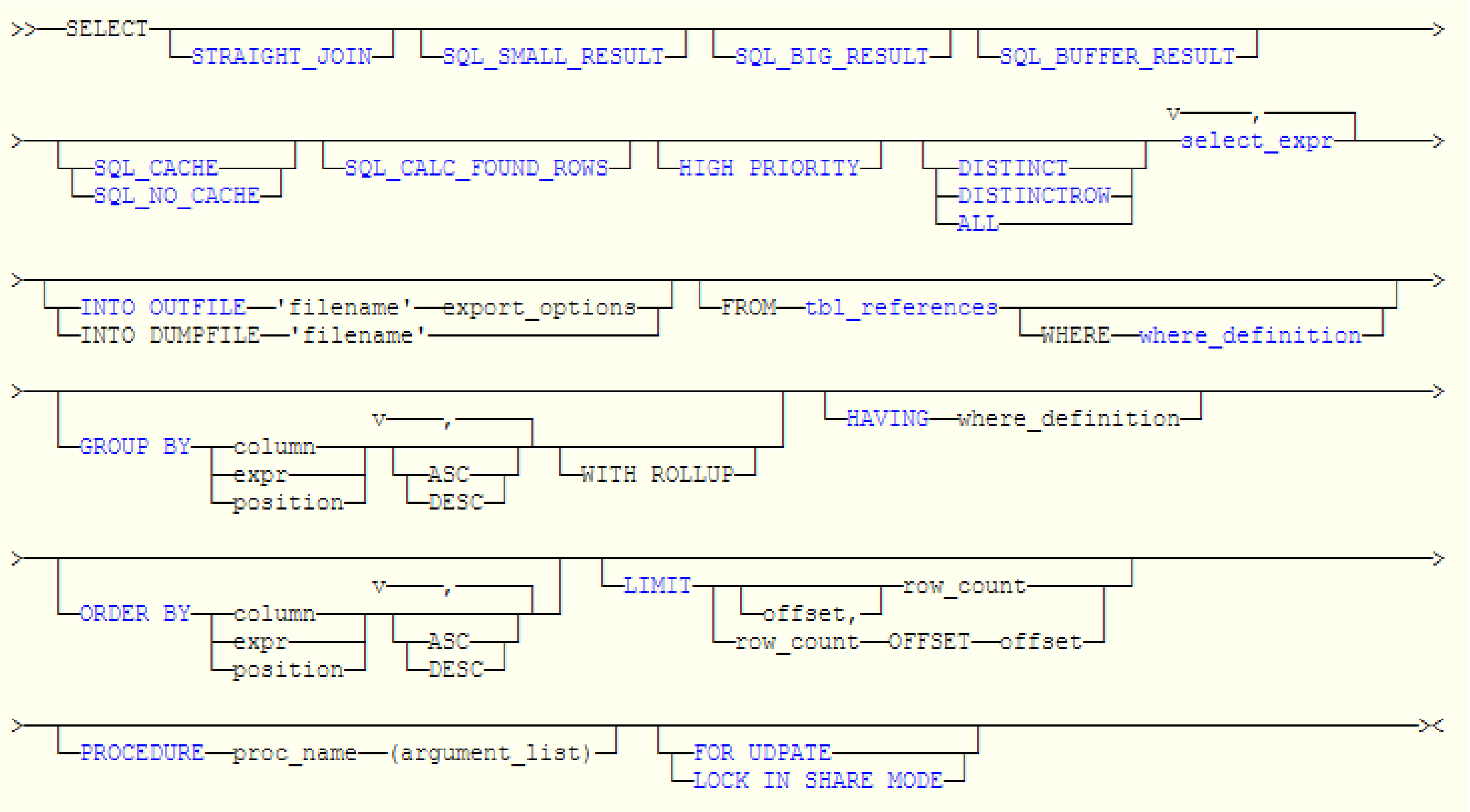
3. **Κέντρο ΠΛΗΝΕΤ Φλώρινας.** *Δημιουργία Database-Driven Web Site.*

4. **Quest Software, Inc.** *Knowledge Xpert for MySQL.*

5. **Ζαχαρής Κ, Κουνιάκης Χ, Κασιμάτης Ν, Μανωλόπουλος Ι, Οικονόμου Θ.** *Βάσεις Δεδομένων.* s.l. : ΟΕΔΒ (Β' Τάξη 1ου Κύκλου ΤΕΕ).

4 Παράρτημα

Πλήρης Σύνταξη SELECT



Πλήρης Σύνταξη DELETE

```
>>─DELETE──┬─LOW PRIORITY─┬─FROM──tbl_name──┬──────────────────┬──────>
           ├─QUICK────────┤                 └─where_definition─┘
           └─IGNORE───────┘

>──┬─────────────────────┬──┬──────────────────┬───────────────────────><
   │ v─────────,───────┐ │  └─LIMIT──row_count─┘
   └─ORDER BY──column──┴─┘
```

Πλήρης Σύνταξη UPDATE

```
                                                    v──────,──────┐
>>──UPDATE──┬──────────────┬─┬────────┬─tbl_name──SET──col_name=value─┴───>
            └─LOW PRIORITY─┘ └─IGNORE─┘

>──┬──────────────┬─┬────────────┬─┬─────────┬─────────────────────><
   └─WHERE clause─┘ │ v────────┐ │ └─LIMIT n─┘
                    └─ORDER BY─┴─┘
```

Πλήρης Σύνταξη INSERT